**NAME:** SREYA G **REGISTER NO.:** 230701334

**CLASS :** CSE F **DATE:** 31/09/2024

**EX-6:**

**COMPETITIVE PROGRAMMING:**

PROBLEM 1:

AIM:

A screenshot of a computer

Description automatically generated

ALGORITHM:

1. Input integer n and array arr[] of size n.

2. Set d = -1 to track duplicate.

3. For each element j, compare it with every subsequent element k.

4. If arr[j] == arr[k], set d = arr[j] and break out of the loops.

5. If a duplicate is found (d != -1), output d; otherwise, print "No duplicates found".

CODE:

#include<stdio.h>

int main()

{

int n;

scanf("%d",&n);

int arr[n];

int d = -1;

for(int i = 0;i < n;i++)

{

scanf("%d",&arr[i]);

}

for(int j = 0;j < n;j++)

{

for(int k = j+1;k < n;k++)

{

if(arr[j] == arr[k])

{

d = arr[j];

break;

}

}

if(d != -1)

{

break;

}

}

if(d != -1)

{

printf("%d",d);

}

else

{

printf("No duplicates found\n");

}

return 0;

}

OUTPUT:

A screenshot of a test

Description automatically generated

RESULT:

Thus the code is executed successfully and gives the expected output.

PROBLEM 2:

AIM:

A screenshot of a computer

Description automatically generated

ALGORITHM:

1. Input integer n and array a[] of size n.

2. For each element i, input the value and store it in a[i].

3. For each element i starting from 1, compare a[i] with a[j].

4. If a match is found, print a[i] and stop.

5. Otherwise, increment j and continue.

CODE:

#include<stdio.h>

int main()

{

int n,c;

int j = 0;

scanf("%d",&n);

int a[n];

for(int i = 0;i <= n;i++)

{

scanf("%d",&c);

a[i] = c;

}

for(int i = 1;i <= n;i++)

{

if(a[j] == a[i])

{

printf("%d",a[i]);

break;

}

else

{

j++;

}

}

}

OUTPUT:

A screenshot of a test

Description automatically generated

RESULT:

Thus the code is executed successfully and gives the expected output.

PROBLEM 3:

AIM:

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

ALGORITHM:

1. Input the number of test cases T.

2. For each test case:

- Input the size N1 and elements of array arr1[].

- Input the size N2 and elements of array arr2[].

3. Initialize two pointers i = 0 and j = 0 for both arrays.

4. While both pointers are within bounds of their respective arrays:

- If arr1[i] == arr2[j], print the value and move both pointers forward.

- If arr1[i] < arr2[j], increment pointer i.

- Otherwise, increment pointer j.

5. Print the intersection of the two arrays.

CODE:

#include<stdio.h>

void intersection(int arr1[],int N1,int arr2[],int N2){

int i = 0,j = 0;

int first = 1;

while(i < N1 && j < N2){

if(arr1[i] == arr2[j]){

if(!first) printf(" ");

printf("%d",arr1[i]);

first = 0;

i++;

j++;

}else if(arr1[i] < arr2[j]){

i++;

}else{

j++;

}

}

printf("\n");

}

int main(){

int T;

scanf("%d",&T);

while(T--){

int N1,N2;

scanf("%d",&N1);

int arr1[N1];

int i;

for(i = 0;i < N1;i++){

scanf("%d",&arr1[i]);

}

scanf("%d",&N2);

int arr2[N2];

for(i = 0;i <N2;i++){

scanf("%d",&arr2[i]);

}

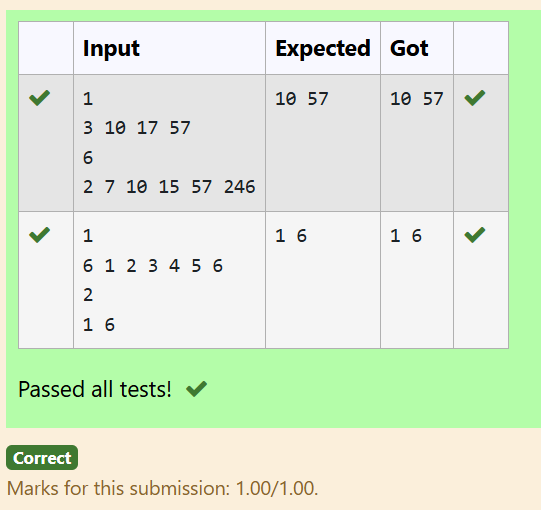
intersection(arr1,N1,arr2,N2);

}

return 0;

}

OUTPUT:



RESULT:

Thus the code is executed successfully and gives the expected output.

PROBLEM 4:

AIM:

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

ALGORITHM:

1. Input number of test cases T.

2. For each test case:

- Input the size N1 and elements of array arr1[].

- Input the size N2 and elements of array arr2[].

3. Initialize two pointers i = 0 and j = 0 to traverse both arrays.

4. While both pointers are within bounds:

- If arr1[i] == arr2[j], print the element and move both pointers forward.

- If arr1[i] < arr2[j], increment pointer i.

- Otherwise, increment pointer j.

5. After processing all test cases, print the intersection of the two arrays for each test case.

CODE:

#include<stdio.h>

void intersection(int arr1[],int N1,int arr2[],int N2){

int i = 0,j = 0;

int first = 1;

while(i < N1 && j < N2){

if(arr1[i] == arr2[j]){

if(!first){

printf(" ");

}

printf("%d",arr1[i]);

first = 0;

i++;

j++;

}else if(arr1[i] < arr2[j]){

i++;

}else{

j++;

}

}

printf("\n");

}

int main(){

int T;

scanf("%d",&T);

while(T--){

int N1,N2;

scanf("%d",&N1);

int arr1[N1];

int i;

for(i = 0;i < N1;i++){

scanf("%d",&arr1[i]);

}

scanf("%d",&N2);

int arr2[N2];

for(i = 0;i < N2;i++){

scanf("%d",&arr2[i]);

}

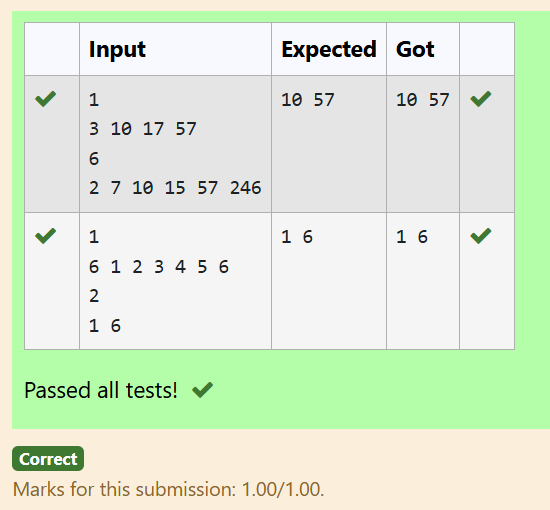
intersection(arr1,N1,arr2,N2);

}

return 0;

}

OUTPUT:

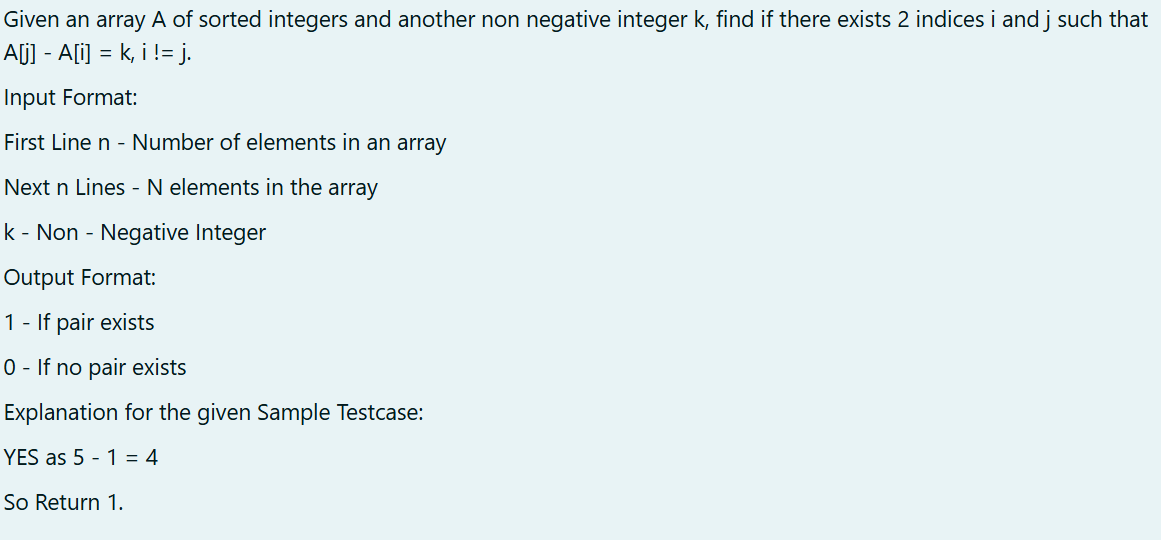


RESULT:

Thus the code is executed successfully and gives the expected output.

PROBLEM 5:

AIM:



A screenshot of a computer

Description automatically generated

ALGORITHM:

1. Input the size of the array n and the difference k.

2. Input the array elements arr[].

3. For each pair of elements arr[i] and arr[j] where i < j:

- If the absolute difference between arr[j] and arr[i] equals k, return 1 (pair found).

- If the difference is greater than k, break the inner loop (optimization).

4. If no such pair is found, return 0.

5. Output the result of the function findPairsWithDiff.

CODE:

#include<stdio.h>

int findPairsWithDiff(int arr[],int n,int k){

int i,j;

for(i = 0;i < n - 1;i++){

for(j = i + 1;j < n;j++){

if(arr[j] - arr[i] == k){

return 1;

}else if(arr[j] - arr[i] > k){

break;

}

}

}

return 0;

}

int main(){

int n,k;

scanf("%d",&n);

int arr[n];

int i;

for(i = 0;i < n;i++){

scanf("%d",&arr[i]);

}

scanf("%d",&k);

printf("%d\n",findPairsWithDiff(arr,n,k));

return 0;

}

OUTPUT:

A screenshot of a computer

Description automatically generated

RESULT:

Thus the code is executed successfully and gives the expected output.

PROBLEM 6:

AIM:

A screenshot of a computer

Description automatically generated

ALGORITHM:

1. Input the size n of the array and the difference k.

2. Input the array elements arr[].

3. Initialize two pointers i = 0 and j = 1.

4. While j < n:

- If arr[j] - arr[i] == k, return 1 (pair found).

- If the difference is less than k, increment j to check the next element.

- If the difference is greater than k, increment i and ensure i < j by adjusting j if needed.

5. If no pair is found, return 0.

6. Output the result from findPairsWithDiff.

CODE:

#include<stdio.h>

int findPairsWithDiff(int arr[],int n,int k){

int i = 0,j = 1;

while(j < n){

if(arr[j] - arr[i] == k){

return 1;

}else if(arr[j] - arr[i] < k){

j++;

}else{

i++;

if(i == j){

j++;

}

}

}

return 0;

}

int main(){

int n,k;

scanf("%d",&n);

int arr[n];

int i;

for(i = 0;i < n;i++){

scanf("%d",&arr[i]);

}

scanf("%d",&k);

printf("%d\n",findPairsWithDiff(arr,n,k));

return 0;

}

OUTPUT:

A screenshot of a computer

Description automatically generated

RESULT:

Thus the code is executed successfully and gives the expected output.